Phase 3 - Implementation and Testing

Submission Deadline: 03rd April 2024 @2000

Submission Guidance: You will submit to turnitin the **hash** of the final commit for the project, `phase\3` will be merged into `main` once you have finished developing and testing.

- Official Documentation for Markdown
- GitHub Documentation for Markdown-Cheatsheet
- C# documentation
- XAML overview (WPF .NET)
- Tutorial: Create a new WPF app with .NET

Structure of the Implementation and Testing Phase Report

You will need to produce a **markdown** documentation in the form of a report that details the requirements phase of the project.

- ensure that once the project is cloned to your local machine that you checkout to `'phase/3'` and branch off of this to your own branch using the following standard: `'phase/3-gitusername'`. As part of your workflow, you will merge back into `'phase/3'` **ONLY**. .
- **Do not** merge to `main`

Working as a team you will assign each other tasks to complete phase 3, refer to the required sections for the `README.md` file below.

The implementation and Testing Phase `README.md` should include the following information:

- **A table indicating** - the allocation of tasks to team members
- **Implementation section** - The programming language you should use is C#. In this section, you have to explain which the main classes and methods are and what their purpose is. You do not need to explain what every single method does – just the core methods and the ones with a more complex logic. Make sure you follow your design when you implement the classes and methods chosen.
- **Testing section** - Explain what kind of testing strategies you have used and why you did choose these strategies. Make sure you test the program thoroughly. Specify what tests you have run and the results you got, e.g. has the system passed the test or not. Use a tabular format. Include also screen shots of the actual tests' execution.
- **Important Note:**
  - Each member of the team should contribute to both implementation and testing.
  - Final submission you must merger `phase/3` into `main`

Marking Scheme

This phase of the assignment contributes 40% of the total coursework mark. The marks are distributed as follows:

- Implementation 65%
- Testing 15%
- Git Workflow 15%
- README.md 5%

Full Rubric

Implementation (65%)

Score Range	Functionality
0-29% Fail	The system fails to implement the specified requirements from the design phase. Essential features of a library management system are missing or not functioning correctly.
30-39% Fail	Limited implementation of the specified requirements. Some essential features are present, but significant improvements are needed for full functionality.
40-49% Satisfactory	Basic implementation of the specified requirements. Essential features are present but lack sophistication in functionality. Improvements are needed for a fully functional system.
50-59% Good	Good implementation of the specified requirements. Essential features are present and functioning correctly, meeting the basic criteria for a library management system.
60-69% Very Good	Very good implementation of the specified requirements. Essential features are impressive and function correctly, providing a high level of functionality for a library management system.
70-79% Excellent	Excellent implementation of the specified requirements. Essential features are exceptional, setting a benchmark for functionality in a library management system.
80-100% Exceptional	Exceptional implementation of the specified requirements. Essential features are outstanding, demonstrating an unparalleled level of functionality for a library management system.

Score Range	Code Quality
0-29% Fail	The code is disorganised and lacks modularity. It does not adhere to best practices and design principles. Naming conventions are inconsistent and do not follow established standards.
30-39% Fail	Limited organisation and modularity in the code. Some adherence to best practices and design principles, but significant improvements are needed. Naming conventions are inconsistent and partially follow established standards.
40-49% Satisfactory	Basic organisation and modularity in the code. Adherence to best practices and design principles is present but lacks sophistication. Naming conventions are basic and partially follow established standards.
50-59% Good	Good organisation and modularity in the code. Adherence to best practices and design principles is above average. Naming conventions are good and mostly follow established standards.
60-69% Very Good	Very good organisation and modularity in the code. Adherence to best practices and design principles is impressive. Naming conventions are very good and consistently follow established standards.
70-79% Excellent	Excellent organisation and modularity in the code, setting a benchmark for best practices and design principles. Naming conventions are excellent and consistently follow established standards.
80-100% Exceptional	Exceptional organisation and modularity in the code, demonstrating an unparalleled level of adherence to best practices and design principles. Naming conventions are exceptional and consistently follow established standards.

Score Range	UI Design
0-29% Fail	The UI is not intuitive and lacks user-friendliness. It does not follow established design principles for WPF, resulting in a poor user experience.
30-39% Fail	Limited intuitiveness and user-friendliness in the UI. The design does not fully adhere to established principles for WPF, leading to a subpar user experience.
40-49% Satisfactory	Basic intuitiveness and user-friendliness in the UI. The design follows minimum requirements for WPF principles but lacks sophistication. Improvements are needed for a better user experience.
50-59% Good	Good intuitiveness and user-friendliness in the UI. The design adheres to established principles for WPF, providing a satisfactory user experience.
60-69% Very Good	Very good intuitiveness and user-friendliness in the UI. The design is impressive and adheres well to established principles for WPF, offering a high-quality user
70-79% Excellent	Excellent intuitiveness and user-friendliness in the UI, setting a benchmark for adherence to WPF principles. The design is excellent, providing an outstanding user
80-100% Exceptional	Exceptional intuitiveness and user-friendliness in the UI, demonstrating an unparalleled level of adherence to WPF principles. The design is exceptional, setting a benchmark for an exceptional user experience.

Testing

Score Range	Description
0-29% Fail	Unit and integration testing are either absent or inadequate, failing to identify and isolate issues. Testing strategy shows a lack of understanding of software testing principles.
30-39% Fail	Limited unit and integration testing present, but they are not comprehensive, and issues are not effectively identified and isolated. Testing strategy lacks depth and may not cover critical parts of the code.
40-49% Satisfactory	Basic unit and integration testing are evident, but improvements are needed for thorough coverage and issue identification. Testing strategy is somewhat effective but lacks sophistication.
50-59% Good	Unit and integration testing are good, covering critical parts of the code. Issues are identified and isolated effectively. Testing strategy is sound and demonstrates a good understanding of software testing principles.
60-69% Very Good	Thorough unit and integration testing are impressive, effectively covering critical parts of the code. Issues are identified and isolated with a high level of competence. Testing strategy is sophisticated and well-executed.
70-79% Excellent	Excellent unit and integration testing showcase a comprehensive approach to identifying and isolating issues. Testing strategy is excellent and demonstrates a deep understanding of software testing principles.
80-100% Exceptional	Exceptional unit and integration testing exhibit an unparalleled level of thoroughness in identifying and isolating issues. Testing strategy is exceptional and sets a benchmark for understanding software testing principles.

Git Workflow(15%)

Score Range	Description
0-29% Fail	Version control using Git is either absent or severely deficient. Commits are unclear and lack meaningful messages. Branching and merging are not effectively utilized, leading to confusion in project development.
30-39% Fail	Limited use of version control with Git. Commits are basic but lack clarity in their messages. Branching and merging are used but may lead to some confusion in project development.
40-49% Satisfactory	Basic version control with Git is evident, but improvements are needed. Commits are clear, but messages may not fully convey the changes. Branching and merging are present but need refinement for a smoother workflow.
50-59% Good	Good version control with Git, with clear and meaningful commits. Branching and merging are effective, contributing to a smooth workflow.
60-69% Very Good	Very good version control with Git, showcasing a high level of clarity in commits. Branching and merging are impressive, ensuring an efficient and organized workflow.
70-79% Excellent	Excellent version control with Git sets a benchmark for clarity in commits. Branching and merging are exceptional, demonstrating an outstanding level of organisation and efficiency in the workflow.
80-100% Exceptional	Exceptional version control with Git is outstanding, with an unparalleled level of clarity in commits. Branching and merging are exceptional, setting a benchmark for an organized and efficient project development workflow.

README.md (5%)

Score Range	Description
0-29% Fail	The README.md is either absent or severely deficient. It lacks structure and fails to provide essential information about the project. Visuals and documentation, if present, do not contribute to understanding.
30-39% Fail	Limited information in the README.md, lacking structure and essential details about the project. Visuals and documentation, if present, provide minimal contribution to understanding.
40-49% Satisfactory	The README.md is basic but lacks sophistication. It provides some information about the project, but improvements are needed for clarity and completeness. Visuals and documentation, if present, contribute to a limited understanding.
50-59% Good	The README.md is good, offering sufficient information about the project. Visuals and documentation, if present, contribute to a good understanding.
60-69% Very Good	The README.md is very good, providing a high level of information about the project. Visuals and documentation, if present, contribute impressively to understanding.
70-79% Excellent	An excellent README.md sets a benchmark for clarity and completeness. Visuals and documentation, if present, contribute exceptionally to understanding the project.
80-100% Exceptional	An exceptional README.md is outstanding, providing unparalleled clarity and completeness. Visuals and documentation, if present, are exceptional, setting a benchmark for understanding the project.